DETAILED ACTION

1. Applicant's amendment dated March 24, 2010, responding to the Final Office action mailed November 24, 2009 provided in the rejection of claims 1-4, 8-14, 18-21, 26, and 31-36, wherein claims 1, 3, 6-8, 10, 11, 13, 15, 16, 18, 20, 21, 23, 25, 26, 29, and 29 have been amended and further claims 2, 4, 5, 12, 14, 17, 22, 24, 27, and 30-36 have been canceled.

Further, in the interest of compact prosecution, the examiner is authorized by Mr. Joseph Lutz (Reg. No. 43,765) on April 9, 2010 in a telephone interview to further amend the claims 1, 3, 11, 13, 21, and 26 (see Examiner's Amendment below) to obviate any potential 35 U.S.C 101 and 112, second paragraph issues and subsequently uses those newly amend claims for further examination.

2. An examiner's amendment to the record appears below. Should the changes and/or additions be unacceptable to applicant, an amendment may be filed as provided by 37 CFR 1.312. To ensure consideration of such an amendment, it MUST be submitted no later than the payment of the issue fee.

3. The application has been amended as follows:

IN THE CLAIMS,

1. (Currently Amended) A method comprising:

configuring one or more processors into a D-stage processor pipeline;

constructing a flow network model for the ~~for the~~ for a sequential network application program;

selecting a plurality of preliminary pipeline stages from the flow network model;

modifying the preliminary pipeline stages to perform control flow and variable transmission therebetween for transforming the sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program; and

executing the D-pipeline stages in parallel within the D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program,

wherein constructing the flow network model comprises:

assigning a unique source node and a unique sink node to the flow network model,

adding a program node to the flow network model for each ~~SSC node~~ strongly-connected component (SSC) node identified in the summary graph of the dependence graph,

adding a variable node to the flow network model for each variable that is defined and used by multiple program nodes,

adding a control node C to the flow network model for each SSC node identified in the summary graph of the dependence graph as a source of control dependence,

generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes,

generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes, and

generating edges between the program nodes and one of the source node and the sink node; and

wherein generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes further comprises:

(i) selecting a program node N that defines a variable node V,

(ii) adding a definition edge from node N to node V with a weight VCost to the flow network model,

(iii) repeating (i) - (ii) for each program node N that defines a variable node V,

(iv) selecting a program node M that uses a variable node W,

(v) adding an edge from the node W to the program node M with an assigned weight of infinity to the flow network model, and

(vi) repeating (iv) - (v) for each program node M that uses a variable node W.

2. (Cancelled)

3. (Currently Amended) The method of claim 1, wherein constructing the flow network model comprises:

transforming the application program into a static, single-assignment form;

building a control flow graph for a loop body of the application program;

building a dependence graph based on a summary graph of the control flow graph and identified, ~~strongly connected components (SSC)~~ SSC nodes of the control flow graph; and

constructing the flow network model according to a summary graph of the dependence graph and identified SSC nodes of the dependence graph.

4-5. (Cancelled)

6. (Previously Presented) The method of claim 1, wherein generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes comprises:

(i) selecting a program node N that has an associated control node C;

(ii) adding a definition edge from the selected node N to the associated control node C;

(iii) associating a weight CCost to the edge;

(iv) repeating (i) - (iii) for each program node that has an associated control node;

(v) selecting a program node N having a controlled dependence on another program node M;

(vi) associating M with the control node C;

(vii) adding an edge from the associated control node C to the selected program node N;

(viii) assigning a weight of infinity to the edge; and

(ix) repeating (v) - (viii) for each node N that has a controlled dependence on another program node M.

7. (Previously Presented) The method of claim 1, wherein generating the edges between program nodes and one of the source node and the sink nodes comprises:

(i) selecting a program node without predecessor node in the flow network model;

(ii) adding an edge from the source node to the selected program node;

(iii) assigning a weight of zero to the edge;

(iv) repeating (i) - (iii) for each program node that has no predecessors;

(v) selecting a program node that has no successors in the flow network;

(vi) adding an edge from the selected program node to the sink node;

(vii) assigning a weight of zero to the added edge; and

(viii) repeating (v) - (vii) for each program node without a successor node in the flow network model.

8. (Previously Presented) The method of claim 1, wherein selecting the plurality of preliminary pipeline stages comprises:

cutting the flow network model into D-1 successive cuts, such that each cut is a balanced minimum cost cut.

9. (Original) The method of claim 8, wherein cutting is performed using an iterative balanced to push-relabel algorithm.

10. (Previously Presented) The method of claim 1, wherein modifying the preliminary pipeline stages comprises:

(a) selecting a preliminary pipeline stage;

(b) altering the selected preliminary pipeline stage to enable proper transmission of live variables and control flow to and from the selected preliminary pipeline stage; and

(c) (a) – (b) for each preliminary pipeline stage to form the D-pipeline stages of a parallel network application.

11. (Currently Amended) An article of manufacture including ~~a machine~~ computer readable storage medium having stored thereon instructions which may be used to program a ~~system~~ computer to perform a method, comprising:

configuring one or more processors into a D-stage processor pipeline;

constructing a flow network model ~~for the~~ for a sequential network application program;

selecting a plurality of preliminary pipeline stages from the flow network model;

modifying the preliminary pipeline stages to perform control flow and variable transmission therebetween for transforming the sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program; and

executing the D-pipeline stages in parallel within the D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program,

wherein constructing the flow network model comprises:

assigning a unique source node and a unique sink node to the flow network model,

adding a program node to the flow network model for each ~~SSC node~~ strongly-connected component (SSC) node identified in the summary graph of the dependence graph,

adding a variable node to the flow network model for each variable that is defined and used by multiple program nodes,

adding a control node C to the flow network model for each SSC node identified in the summary graph of the dependence graph as a source of control dependence,

generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes,

generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes, and

generating edges between the program nodes and one of the source node and the sink node; and

wherein generating the edges between program nodes and one of the source node and the sink nodes comprises:

(i) selecting a program node without predecessor node in the flow network model,

(ii) adding an edge from the source node to the selected program node,

(iii) assigning a weight of zero to the edge,

(iv) repeating (i) - (iii) for each program node that has no predecessors,

(v) selecting a program node that has no successors in the flow network,

(vi) adding an edge from the selected program node to the sink node,

(vii) assigning a weight of zero to the added edge, and

(viii) repeating (v) - (vii) for each program node without a successor node in the flow network model.

12. (Cancelled)

13. (Currently Amended) The article of manufacture of claim 11, wherein constructing the flow network model comprises:

transforming the application program into a static, single-assignment form;

building a control flow graph for a loop body of the application program;

building a dependence graph based on a summary graph of the control flow graph and identified, ~~strongly-connected components (SSC)~~ SSC nodes of the control flow graph; and

constructing the flow network model according to a summary graph of the dependence graph and identified SSC nodes of the dependence graph.

14. (Cancelled)

15. (Previously Presented) The article of manufacture of claim 11, generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes further comprises:

(i) selecting a program node N that defines a variable node V;

(ii) adding a definition edge from node N to node V with a weight VCost to the flow network model;

(iii) repeating (i) - (ii) for each program node N that defines a variable node V;

(iv) selecting a program node M that uses a variable node W;

(v) adding an edge from the node W to the program node M with an assigned weight of infinity to the flow network model; and

(vi) repeating (iv) - (v) for each program node M that uses a variable node W.

16. (Previously Presented) The article of manufacture of claim 11, wherein generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes comprises:

(i) selecting a program node N that has an associated control node C;

(ii) adding a definition edge from the selected node N to the associated control node C;

(iii) associating a weight CCost to the edge;

(iv) repeating (i) - (iii) for each program node that has an associated control node;

(v) selecting a program node N having a controlled dependence on another program node M;

(vi) associating M with the control node C;

(vii) adding an edge from the associated control node C to the selected program node N;

(viii) assigning a weight of infinity to the edge; and

(ix) repeating (v) - (viii) for each node N that has a controlled dependence on another program node M.

17. (Cancelled)

18. (Previously Presented) The article of manufacture of claim 11, wherein selecting the plurality of preliminary pipeline stages comprises:

cutting the flow network model into D-1 successive cuts, such that each cut is a balanced minimum cost cut.

19. (Original) The article of manufacture of claim 18, wherein cutting is performed using an iterative balanced to push-relabel algorithm.

20. (Previously Presented) The article of manufacture of claim 11, wherein modifying the preliminary pipeline stages comprises:

selecting a preliminary pipeline stage;

altering the selected preliminary pipeline stage to enable proper transmission of live variables to and from the selected preliminary pipeline stage;

altering the selected preliminary pipeline stage to enable proper transmission of control flow to and from the selected preliminary pipeline stage; and

repeating the selecting, altering and altering for each preliminary stage to form the D-pipeline stages of a parallel network application.

21. (Currently Amended) A computer-implemented method comprising:

constructing a flow network model from a sequential network application program;

cutting the flow network model into a plurality of preliminary pipeline stages; and

transforming the preliminary pipeline stages to perform control flow and variable transmission therebetween to form D-pipeline stages that collectively

perform an infinite packet processing stage (PPS) loop of the sequential network application program to enable parallel execution of the infinite PPS loop of the sequential network application program,

wherein transforming the preliminary application program stages comprises:

(i) electing a preliminary application program stage,

(ii) selecting a control flow graph generated for the infinite PPS loop corresponding to the selected preliminary application program stage,

(iii) removing instructions from the control flow graph if the instruction is not contained within the selected preliminary pipeline stage,

(iv) transforming the selected control flow graph according to variables and control objects transmitted from the prior stage,

(v) reconstructing the PPS loop from the transformed control flow graph to form a pipeline stage, and

repeating (i) - (v) for each preliminary pipeline stage to form D-pipeline stages of a parallel network application program; and

wherein transforming the selected control flow further comprises:

selecting values for variables that are transmitted from a prior pipeline stage, and

for each variable transmitted to a next pipeline stage, setting a value of the variable to a distinctive temporary following definition of the variable within the control flow graph.

22. (Cancelled)

23. (Previously Presented) The method of claim 21, wherein transforming the control flow further comprises:

selecting values for control objects transmitted from a prior pipeline stage on entry to the control flow graph;

for each control object received from the prior pipeline stage, constructing a conditional instruction using the control object; and

replacing corresponding conditional nodes within the CFG with the conditional instruction.

24. (Cancelled)

25. (Previously Presented) The method of claim 21, wherein transforming the control flow graph further comprises:

for each control object to be transmitted to a next pipeline stage, placing an alternate value of the control object in each alternate successor node of a conditional node associated with the control object in the control flow graph; and

transmitting live set data to a next pipeline stage at exit of the control flow graph.

26. (Currently Amended) An article of manufacture including a ~~machine~~ computer readable storage medium having stored thereon instructions which may be used to program a ~~system~~ computer to perform a method, comprising:

constructing a flow network model from a sequential network application program;

cutting the flow network model into a plurality of preliminary pipeline stages; and

transforming the preliminary pipeline stages to perform control flow and variable transmission therebetween in order to form D-pipeline stages that collectively perform an infinite_packet processing stage (PPS) loop of the

sequential network application program to enable parallel execution of the infinite PPS loop of the sequential network application program,

wherein transforming the preliminary application program stages comprises:

(i) electing a preliminary application program stage,

(ii) selecting a control flow graph generated for the infinite PPS loop corresponding to the selected preliminary application program stage,

(iii) removing instructions from the control flow graph if the instruction is not contained within the selected preliminary pipeline stage,

(iv) transforming the selected control flow graph according to variables and control objects transmitted from the prior stage,

(v) reconstructing the PPS loop from the transformed control flow graph to form a pipeline stage, and

repeating (i) - (v) for each preliminary pipeline stage to form D-pipeline stages of a parallel network application program; and

wherein transforming the selected control flow graph further comprises:

for each control object to be transmitted to a next pipeline stage, placing an alternate value of the control object in each alternate successor node of a conditional node associated with the control object in the control flow graph, and

transmitting live set data to a next pipeline stage at exit of the control flow graph.

27. (Cancelled)

28. (Previously Presented) The article of manufacture of claim 26, wherein transforming the selected control flow graph further comprises:

selecting values for control objects transmitted from a prior pipeline stage on entry to the control flow graph;

for each control object received from the prior pipeline stage, constructing a conditional instruction using the control object; and

replacing corresponding conditional nodes within the control flow graph with the conditional instruction.

29. (Previously Presented) The article of manufacture of claim 26, wherein transforming the selected control flow graph further comprises:

selecting values for variables that are transmitted from a prior pipeline stage; and

for each variable transmitted to a next pipeline stage, setting a value of the variable to a distinctive temporary following definition of the variable within the control flow graph.

30-36. (Cancelled)

- **END OF AMENDMENT –**

Allowable Subject Matter

4. Claims 1, 3, 6-11, 13, 15, 16, 18-21, 23, 25, 26, 28, and 29 (renumbered as 1-20) are allowed.

5. The following is an examiner's statement of reasons for allowance:

The cited prior art taken alone or in combination fails to at least suggest "...

configuring ... into a D-stage processor pipeline;

constructing a flow network model for a sequential network application program;

selecting ... preliminary pipeline stages ...;

modifying the preliminary pipeline stages to perform control flow and variable transmission therebetween for transforming the sequential network application program into D-pipeline stages that collectively perform an infinite packet processing stage (PPS) loop of the sequential network application program; and

executing the D-pipeline stages in parallel within the D-stage processor pipeline to provide parallel execution of the infinite PPS loop of the sequential network application program,

wherein constructing the flow network model comprises:

assigning a unique source node and a unique sink node to the flow network model,

adding a program node to the flow network model for each strongly-connected component (SSC) node identified in the summary graph of the dependence graph,

adding a variable node to the flow network model for each variable that is defined and used by multiple program nodes,

adding a control node C to the flow network model for each SSC node identified in the summary graph of the dependence graph as a source of control dependence,

generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes,

generating edges having an associated weight to connect corresponding program nodes to corresponding control nodes, and

generating edges between the program nodes and one of the source node and the sink node; and

wherein generating edges having an associated weight to connect corresponding program nodes to corresponding variable nodes further comprises:

(i) selecting a program node N that defines a variable node V,

(ii) adding a definition edge from node N to node V with a weight VCost to the flow network model,

(iii) repeating (i) - (ii) for each program node N that defines a variable node V,

(iv) selecting a program node M that uses a variable node W,

(v) adding an edge from the node W to the program node M with an assigned weight of infinity to the flow network model, and

(vi) repeating (iv) - (v) for each program node M that uses a variable node W.", as recited in independent claims 1;

"... generating the edges between program nodes and one of the source node and the sink nodes comprises:

(i) selecting a program node without predecessor node in the flow network model,

(ii) adding an edge from the source node to the selected program node,

(iii) assigning a weight of zero to the edge,

(iv) repeating (i) - (iii) for each program node that has no predecessors,

(v) selecting a program node that has no successors in the flow network,

(vi) adding an edge from the selected program node to the sink node,

(vii) assigning a weight of zero to the added edge, and

(viii) repeating (v) - (vii) for each program node without a successor node in the flow network model" as recited in independent claims 11; and

"... transforming the preliminary application program stages comprises:

(i) electing a preliminary application program stage,

(ii) selecting a control flow graph generated for the infinite PPS loop corresponding to the selected preliminary application program stage,

(iii) removing instructions from the control flow graph if the instruction is not contained within the selected preliminary pipeline stage,

(iv) transforming the selected control flow graph according to variables and control objects transmitted from the prior stage,

(v) reconstructing the PPS loop from the transformed control flow graph to form a pipeline stage, and

repeating (i) - (v) for each preliminary pipeline stage to form D-pipeline stages of a parallel network application program; and

wherein transforming the selected control flow further comprises:

selecting values for variables that are transmitted from a prior pipeline stage, and

for each variable transmitted to a next pipeline stage, setting a value of the variable to a distinctive temporary following definition of

the variable within the control flow graph." as recited in independent claims 21 and similarly recited in independent claim 26 respectively.

6. Thus, all remaining dependent claims (3, 6-10), (13, 15, 16, 18-20), (23, 25) and (28, 29) are considered allowable by virtue of their dependence on allowable independent claims 1, 11, 21, and 26 respectively.

7. Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Conclusion

8. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is (571) 270-1240. The examiner can normally be reached on 8:00-5:30 (EST/EDT), Monday through Friday.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public

PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Ben C Wang/
Examiner, Art Unit 2192

/Michael J. Yigdall/
Primary Examiner, Art Unit 2192